IT314 - Software Engineering

Software Requirements Specifications

Parking Management System

Group No. 29

Group Members

202201431 Heet Thakkar 202201407 Herik Patel

202201417 Vats Shah 202201409 Ramya Shah

202201418 Dev Soni 202201401 Vraj Patel

202201448 Neel Vasoya 202201436 Stavan RaviSaheb

202201415 Darshan Jogadiya

Under the guidance of

Professor: Dr. Saurabh Tiwari Mentor: Jaydeep

INDEX

1. Introduction

The Software Requirements Specification (SRS) introduction overviews the entire SRS, including its purpose, scope, definitions, acronyms, abbreviations, references, and a general overview of the SRS. This document aims to thoroughly analyze and provide a detailed understanding of the **Parking Management System** by defining the problem statement. Additionally, the document focuses on the capabilities required by stakeholders and their needs while outlining the high-

level features of the product. More detailed requirements for the **Parking Management System** are provided later in the document.

1.1 Purpose

This document aims to gather and analyze various ideas for defining the system and its customer requirements. It also aims to predict and organize the product's usage to understand the project better. This document outlines concepts that may be developed later and documents ideas that are being considered but may still need to be implemented.

This SRS document provides a detailed overview of our software product, including its parameters and goals. It describes the target audience, user interface, software requirements, and how the client, team, and audience perceive the product and its functionality. This document also aids designers and developers in the software delivery lifecycle (SDLC) processes.

1.2 Introduction to the Parking Management System

1.2.1 Description of Problem

In urban areas with limited parking, finding a spot can be timeconsuming and frustrating, contributing to traffic congestion, pollution, and stress for drivers. Parking lot owners, on the other hand, often struggle to optimize space utilization and occupancy rates. Our Parking Management System aims to bridge this gap by providing a centralized platform where drivers can locate and pre-book available parking spaces in their vicinity, while parking lot operators can track real-time occupancy, set dynamic pricing, and manage availability based on demand. This system not only helps drivers save time and reduce congestion but also empowers parking lot owners with data-driven insights for effective space and resource management.

1.2.2 Features of the Project (Solution)

- Authorized access

1.3 Scope

The scope of this Parking Management System project is to simplify parking for drivers and enhance management for parking lot owners. It enables drivers to view, book, and pay for parking spots in real-time,

reducing search time and congestion. For lot owners, it provides automated tracking, dynamic pricing, and data analytics to optimize space usage and revenue. This system aims to create a more efficient, user-friendly parking experience while promoting sustainable practices and reducing traffic in urban areas.

1.4 Abbreviation

Throughout the document Parking Management System will be denoted as PMS.

2. Specific Requirements

2.1 Functional Requirements

1. **User Registration and Authentication**:
   - Allows users to create an account with secure credentials (e.g., email or phone verification).
   - Provides login and password recovery options to ensure easy and secure access.
2. **Real-Time Parking Space Availability**:
   - Displays updated availability of parking spots based on real-time data.
   - Allows users to filter parking options by location, distance, and availability status.
3. **Reservation System**:
   - Enables users to pre-book parking spaces with options for date, time, and location.
   - Provides the flexibility to modify or cancel reservations based on availability.
4. **Payment Processing**:
   - Integrates secure payment options, including credit/debit cards and digital wallets.
   - Supports auto-billing and payment history tracking for convenient future bookings.

5. **Booking Confirmation**:

- Sends instant confirmation via app notification, SMS, or email, including booking details.
- Provides a digital receipt with all transaction and reservation details for reference.

6. **Admin Dashboard**:
   - Offers a centralized view for administrators to monitor parking lot occupancy and reservations.
   - Displays revenue data, booking trends, and user activity for effective management.
7. **Parking Instructions and Reminders**:
   - Provides GPS-based directions to guide users to their reserved parking spots.
   - Sends reminders about approaching reservation times and expiry notifications to prevent overstay.
8. **Loyalty Program**:
   - Awards points or discounts for frequent bookings to encourage user retention.
   - Offers users the ability to track loyalty points and redeem them for future bookings or discounts.

2.2 Non-functional Requirements

1. Performance: The system should respond quickly to user queries and load data efficiently.

2. Reliability: Ensure high uptime and data reliability to avoid disruptions in crop storage and retrieval.

3. Scalability: The system should be able to accommodate more users and warehouses as the platform grows.

4. Usability: The user interface should be intuitive and user-friendly for both farmers and warehouse owners.

5. Data Privacy: Protect user information and comply with applicable data privacy laws.

6. Availability: Ensure that the system is available 24/7, with minimal downtime for maintenance.

7. Data Backup: Regularly backup data to prevent data loss in case of system failures.

8. Scalable Database: Use a scalable and robust database system to store user and inventory data.

9. Mobile Accessibility: Ensure that the system is accessible and userfriendly on mobile devices.

10. Cost-Effective: Develop and maintain the system in a cost-effective manner to minimize operational expenses.

3. Product Backlog

Sprint 1: Set up a working environment, UI/UX, Login/Sign-Up

1. As a Parking Lot Owner, I should be able to register into the system with details like name, contact information, and parking lot location so that I can manage my parking facility.
2. As a Parking Lot Owner, I should be able to log in to the system to manage and update my parking lot details.
3. As a Parking Lot User, I should be able to register into the system with details like name, email, and contact number so I can reserve a parking slot. 4. As a Parking Lot User, I should be able to log in to the system to access parking features.
5. As Security Personnel, I should be able to log in to the system so that I can monitor parking operations and verify user entries and exits.

Sprint 2: Booking and Reservation, Parking Instructions

1. As a Parking Lot User, I should be able to view available parking slots so that I can book a slot conveniently.
2. As a Parking Lot User, I want to book a parking slot and receive a booking confirmation to secure my parking space upon arrival.
3. As a Parking Lot User, I should receive instructions on how to locate my reserved parking slot for ease of access.
4. As Security Personnel, I should be able to view the status of reserved parking slots to assist users when necessary.
5. As a Parking Lot User, I want to cancel or modify my booking if needed to manage my schedule better.

Sprint 3: Entry and Exit Management, Notifications (Email/OTP Inclusion)

1. As a Parking Lot User, I want to receive an entry reminder notification before my reservation begins to ensure timely parking.
2. As a Parking Lot User, I should be able to enter the parking lot by validating my reservation, so I don't face any delays.
3. As Security Personnel, I should be able to verify user entry and exit with OTP or booking confirmation to ensure only authorized users access the lot.
4. As a Parking Lot User, I should receive notifications upon successful entry and exit for record-keeping.
5. As Security Personnel, I should receive alerts for any unauthorized entry attempts to maintain security.

Sprint 4: Accessibility Features, Slot Extension, and Payment Integration

1. As a Disabled Parking Lot User, I want to access dedicated parking spots and assistance to park easily and without accessibility issues.
2. As a Parking Lot User, I should be able to extend my parking slot timing if necessary, ensuring I can stay longer without needing a new reservation.

3. As a Parking Lot User, I want to make secure payments within the system to complete my reservation or exit smoothly.
4. As a Parking Lot User, I should receive a receipt immediately after completing the payment, ensuring I have a record.
5. As Security Personnel, I want to assist users in handling transaction failures, so they can complete payments quickly if issues arise.

Sprint 5: Dashboard for Owners, Security Monitoring

1. As a Parking Lot Owner, I want a dashboard displaying real-time data on slots occupied, revenue, and reservations to monitor my parking lot's performance.
2. As a Parking Lot Owner, I should be able to export data from the dashboard into a report format to analyze it further.
3. As Security Personnel, I want access to logs showing parking lot activity, so I can track user entries and exits effectively.
4. As Security Personnel, I should be able to monitor and verify all parking operations, including booking, entrance, and exit actions to ensure smooth functioning.
5. As a Parking Lot Owner, I want the system to notify me of any unauthorized actions or discrepancies so I can take immediate corrective measures.

4. Proposed Model for Development

Agile (Waterfall Incremental) software development is a model that emphasizes collaboration, flexibility, and rapid iteration. It's well suited to the problem statement of designing a system for warehouses and farmers, as the requirements may change and evolve as the development progresses. Here's how the agile model can be used for this problem statement:

- Requirements Gathering: The development team will work closely with stakeholders, such as farmers and warehouse managers, to gather requirements and understand the needs of the system. This may involve conducting user interviews, creating prototypes, and conducting user testing.

- Sprint Planning: The development team will break down the requirements into small, manageable tasks and prioritize them based on importance and dependencies. These tasks will be organized into sprints, or short development cycles, that typically last two to four weeks.
- Development and Testing: During each sprint, the development team will work on completing the tasks assigned to them. This may involve writing code, conducting unit tests, and performing integration testing. The development team will also regularly conduct code reviews to ensure the quality of the code and catch any potential issues early on.
- Sprint Review: At the end of each sprint, the development team will present their progress to stakeholders and receive feedback. This is an opportunity for stakeholders to provide feedback and suggest any changes that need to be made.
- Sprint Retrospective: After the sprint review, the development team will reflect on the sprint and identify areas for improvement. This may involve changes to processes, tools, or the development team's composition.
- Repeat: The process will repeat, with the development team using the feedback and insights from the sprint review to make improvements and move forward. This iteration continues until the system is fully developed and meets the requirements of the stakeholders.

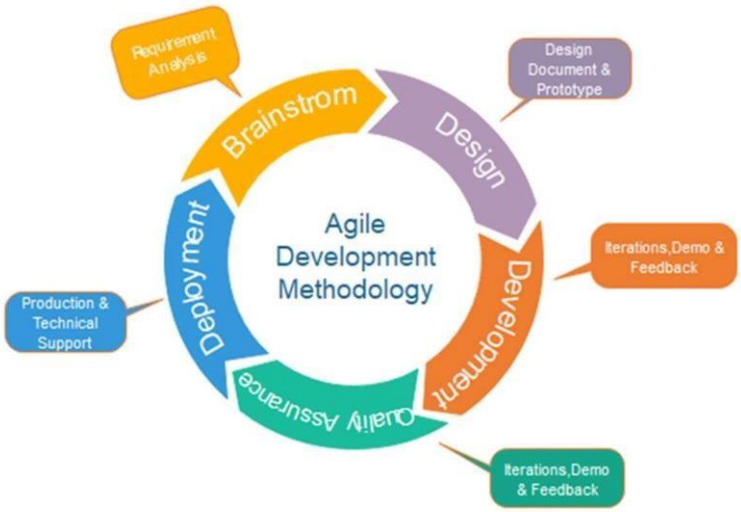

Fig. Agile Model

The agile model allows for frequent feedback and iteration, making it ideal for the problem statement of designing a system for warehouses and farmers. By breaking down the development into small sprints and working closely with stakeholders, the development team can ensure that the system is developed to meet the needs of the users and is delivered in a timely and efficient manner.

5. Technology Used

The PMS is crafted to enhance and streamline warehouse operations. Developed using modern technologies, it leverages HTML, CSS, and ReactJS for an intuitive and responsive frontend. The backend is powered by Python, Django, and SQLite, ensuring robust functionality and data management.

In the PMS, Visual Studio Code (VSCode) serves as the integrated development environment (IDE). VSCode's features, including debugging and source control integration, accelerate development. Github is employed for version control, enhancing collaboration.

6. Assumptions

- All users and parking lot owners have access to compatible smartphone with proper, stable internet connection.
- All the users do have email ID.
- All the invoice generated have their transactions get done between parking lot owner and users and so transaction is not fraudulent. • All the information shown by warehouse owner are correct and up-to-date.

7. References

1. StackOverflow - h ps://stackoverflow.com/
2. GeeksForGeeks - h ps://www.geeksforgeeks.org/
3. ChatGPT - h ps://chat.openai.com/
4. 4. Image of Agile SDLC Model https://static.javatpoint.com/tutorial/software-engineering/images/software-engineering-agile-model.png